Элементы технического зрения

РАСПОЗНАВАНИЕ ИЗОБРАЖЕНИЙ И ИХ ЭЛЕМЕНТОВ. ЛИНИИ НА ИЗОБРАЖЕНИИ

Алгоритм Canny

- Размыть изображение фильтром Гаусса с некоторым σ
  - Убрать шум, лишние детали текстуры
- Рассчитать градиент изображения
  - Одним из операторов – например, Собеля
- Все пиксели где сила краев $< T$ убрать из рассмотрения
- Поиск локальных максимумов
- Прослеживание краев из точек локальных максимумов

Canny - результат

исходное

Canny с $\sigma = 1$

Canny с $\sigma = 2$

Морфологические операции

После построения карты краёв мы можем её бинаризовать

Как бороться с «толстыми» линиями и мелким мусором?

Морфологические операции

Напомню – ранговый фильтр

ранговый фильтр – больше заданного порога

разные пороги для нулей и единиц!

Он же процентильный

1	1	0
1	0	1
1	1	0

- Единиц 6, нулей 3
- Ранг 6 – 1, ранг 7 – 0
- Нужен, если мы имеем априорную информацию

Морфологические операции

Две морфологические операции – расширение и сжатие (dilate и erode)

Сжатие – ранговый фильтр минимального ранга

Расширение - максимального

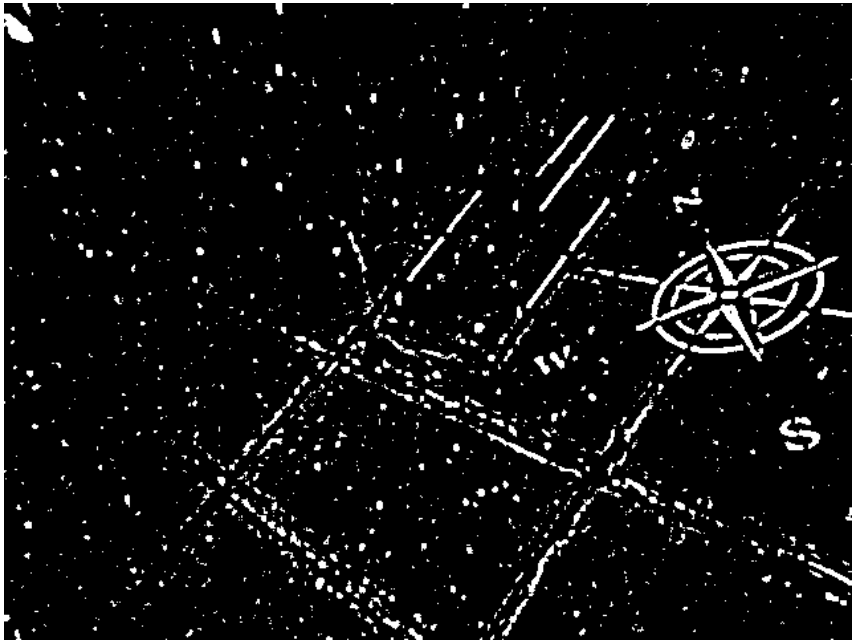
1	1	0
1	0	1
1	1	0

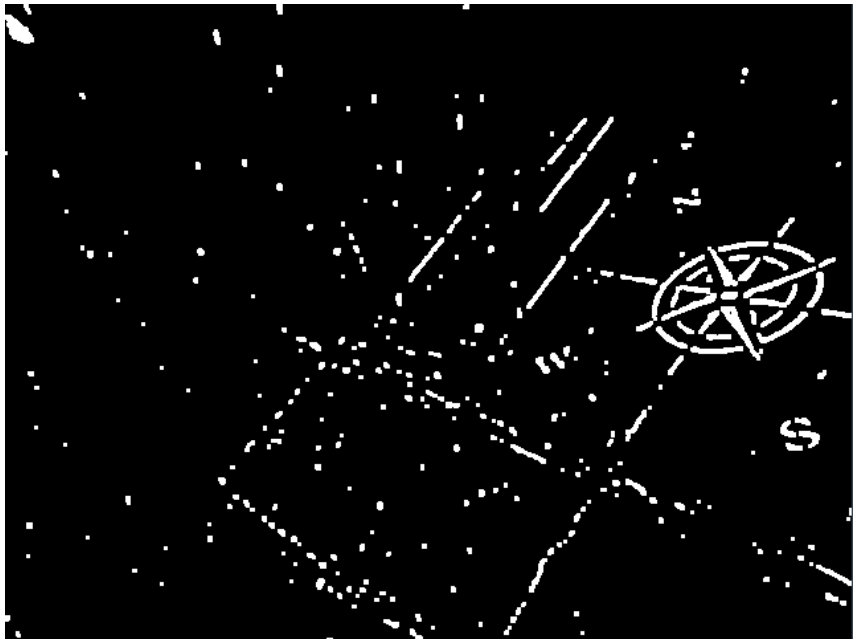
- В данном случае минимальный ранг (нулевой) – 0
- Максимальный (девятый) ранг - 1

Морфологические операции

```
void adaptiveTreshold( int, void* )
{
    cv::adaptiveThreshold(srcGrey, binary, 255, type, cv::THRESH_BINARY, sizeValue*2+3, constValue-11);
    cv::erode(binary, morph, cv::Mat());
    cv::dilate(morph, morph, cv::Mat());
    cv::imshow( window_name, binary );
    cv::imshow(morph_name, morph);
}
```

Морфологические операции





Морфологические операции

Сочетание операций сужения и расширения – операция открытия – позволяет убирать одиночные «белые» пикселы

Можно в обратном порядке – получим операцию закрытия

1	1	0
1	0	1
1	1	0

Морфологические операции

Как уменьшить толщину контура?

Построение скелета объектов

Простейший вариант – итерационное применение морфологических операций

Выберем крестообразный шаблон

1	1	0
1	0	1
1	1	0

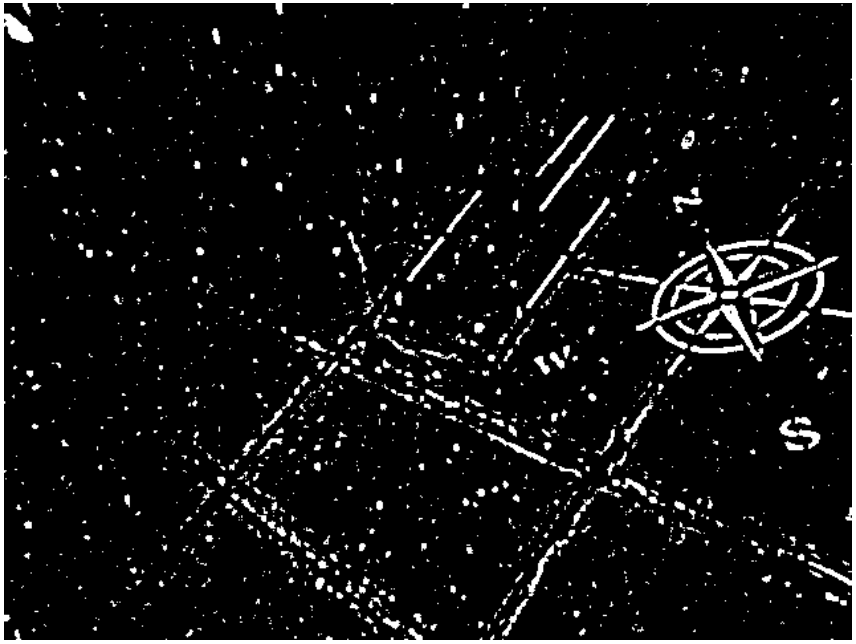
Морфологические операции

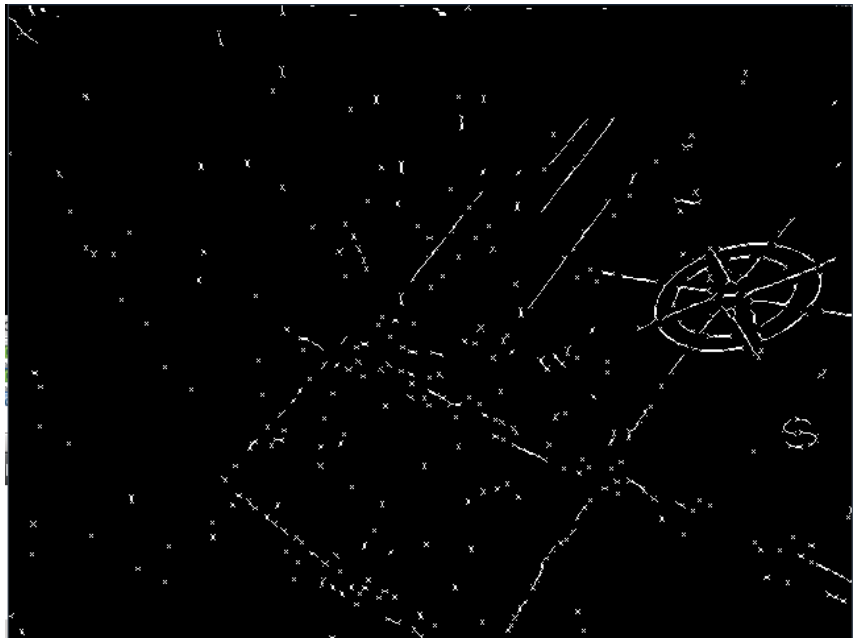
```
cv::Mat thinner (cv::Mat& source)
{
    cv::Mat element = cv::getStructuringElement(cv::MORPH_CROSS, cv::Size(3,3)), eroded, dilated;
    cv::Matlb ret(source.size()), temp(source.size());
    int count = 0, done = 0;

    do
    {
        cv::erode(source, eroded, element);
        cv::dilate(eroded, dilated, element);
        cv::subtract(source, dilated, temp);
        cv::bitwise_or(ret,temp,ret);
        eroded.copyTo(source);
        done = cv::countNonZero(eroded);
        count++;
    }
    while ((count<10)&&(done));
    return ret;
}
```

Opencv_morphology

Морфологические операции





Opencv_morphology

Вопрос

?Получив контур объекта (связный набор пикселей) – как его дальше анализировать?

!Вариант - нужно преобразовать контур в некоторое численное представление

- Один из способов – использование цепных кодов

Работа с контурами

Полигональная аппроксимация

Цепные коды

Дескрипторы контуров

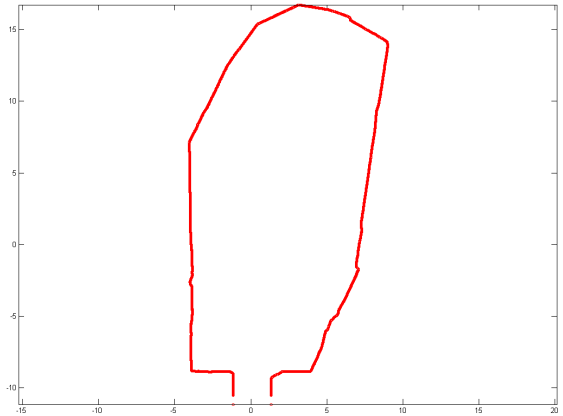
Полигональная аппроксимация

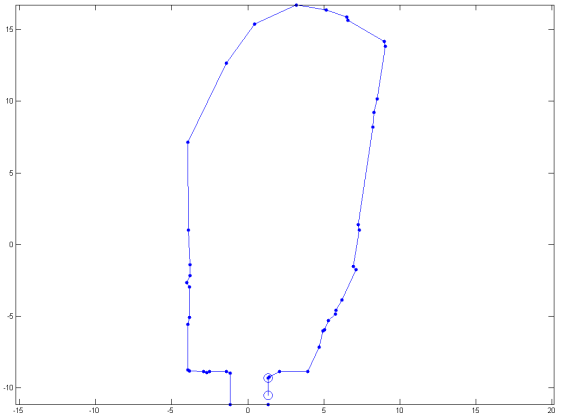
Постановка:

- Аппроксимация точечной кривой ломаной линией

Цель:

- Сжатие информации
- Борьба с дискретностью и шумом
- Облегчение дальнейшего анализа





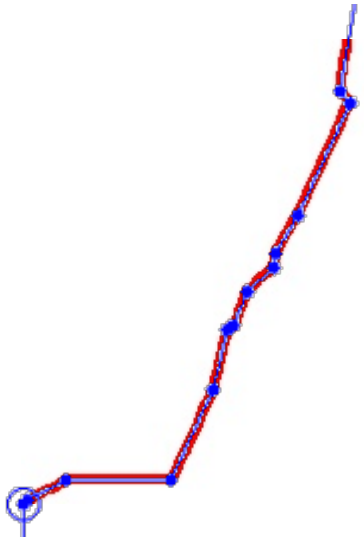
Полигональная аппроксимация

Постановка:

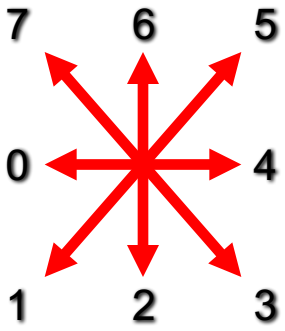
- Аппроксимация точечной кривой ломаной линией

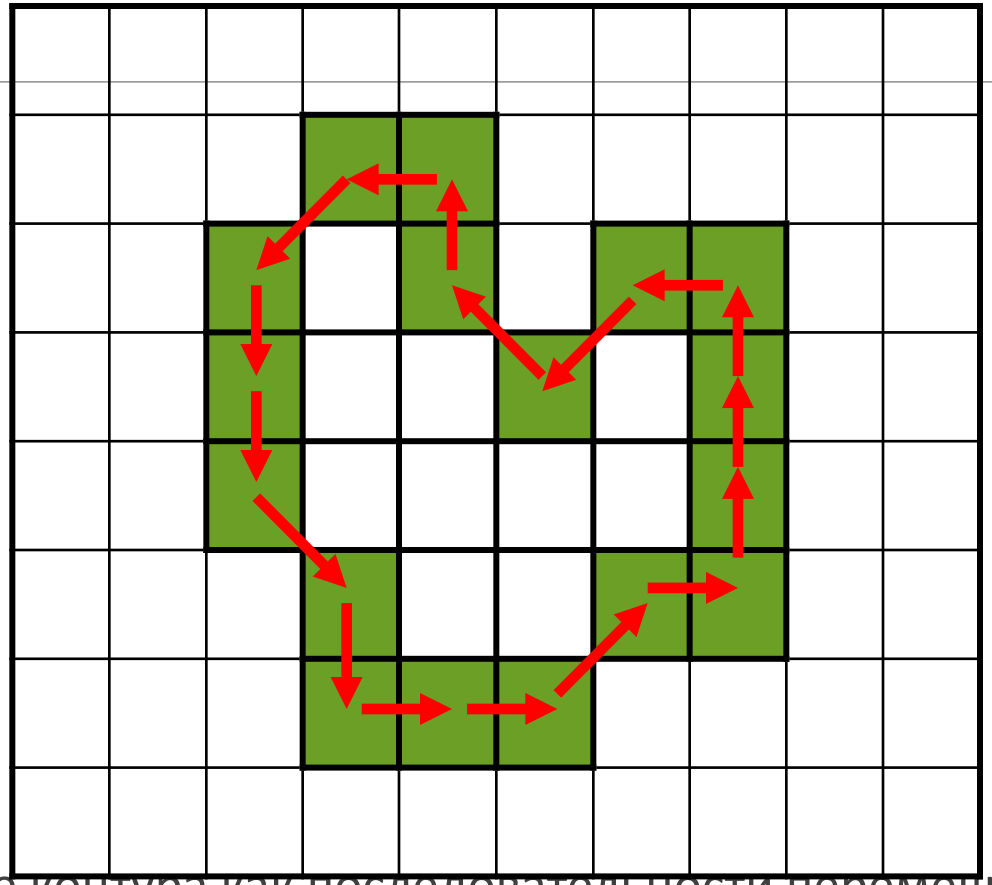
Цель:

- Сжатие информации
- Борьба с дискретностью и шумом
- Облегчение дальнейшего анализа



Цепной код – 8-ми связные контура

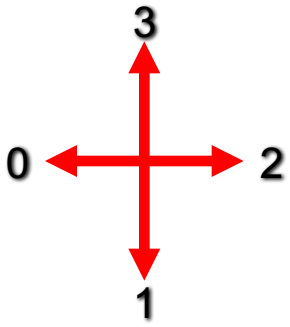


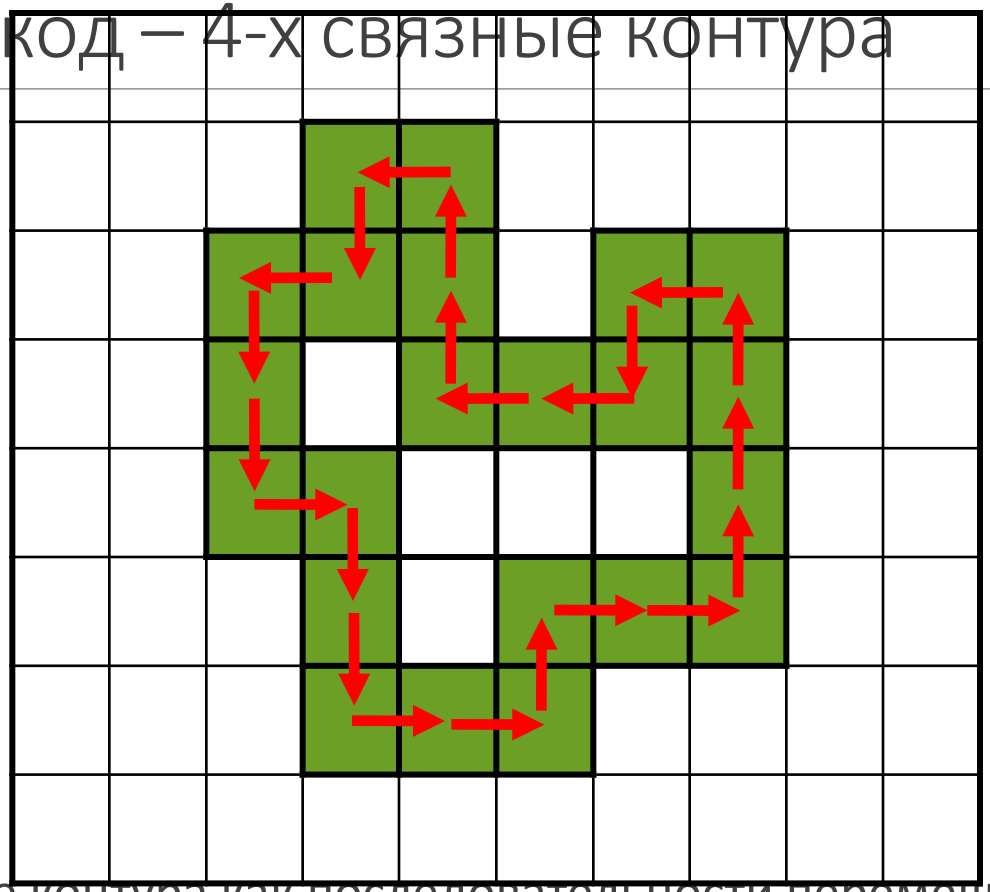


Кодирование контура как последовательности перемещений

Код: 12232445466601760

Цепной код – 4-х связные контура





Кодирование контура как последовательности перемещений

Код: 11223223330100330101112

Цепной код - свойства

Свойства

- Цепной код – представление контура, независимое к его перемещению
- При замене в 8-ми связном кода любого n но (n mod 8) + 1 контур будет **повернут** по часовой стрелке на 45 градусов
- Некоторые особенности контуров, такие как уголки, например могут быть сразу рассчитаны по анализу цепных кодов

Сложности

- В цепном коде важна начальная точка – при ее изменении меняется и код
- Небольшие вариации границы (шум) серьезно меняют код. Сравнение двух шумных контуров по цепному коды – сложно
- Цепной код не инвариантен к повороту

Разностный код

Это «производная» цепного кода

- Формула
  - $y_i = (x_{i+1} - x_i) \bmod 8$ – восьмисвязный
  - $y_i = (x_{i+1} - x_i) \bmod 4$ – четырехсвязный

Разностный код

Свойства:

- Инвариантен к повороту кратному 45 градусам (восьмисвязный)

Проблемы:

- Также чувствителен к шуму
- Не инвариантен к повороту на произвольный угол

Кривизна

Кривизна (*curvature*) – производная $\psi(s)$

- Аналогично разностному цепному коду, но со знаком

$$K(s) = \psi(s) - \psi(s-1)$$

- Представление, инвариантное к повороту и переносу

Функция плотности кривизны

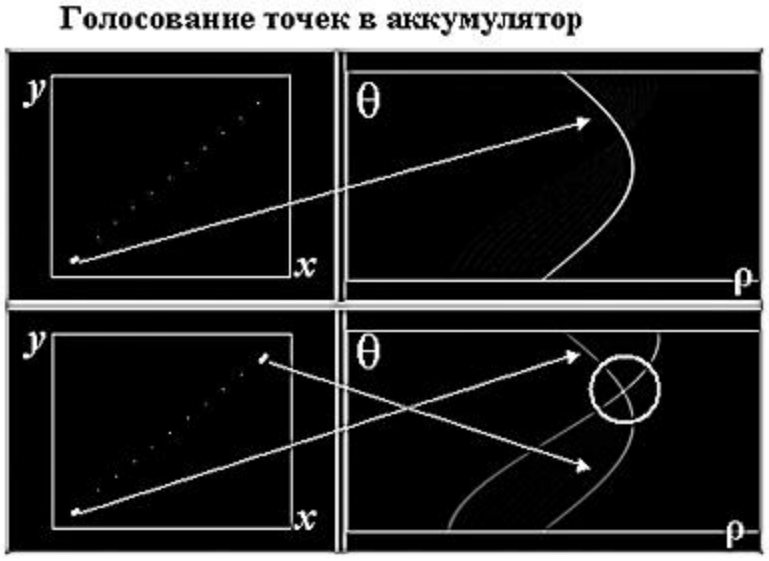
- Гистограмма распределения значений кривизны

Преобразование Хафа

Поиск прямых и окружностей на изображении

Отличие – построение аккумуляторов

$X\cos\theta+Y\sin\theta=\rho.$

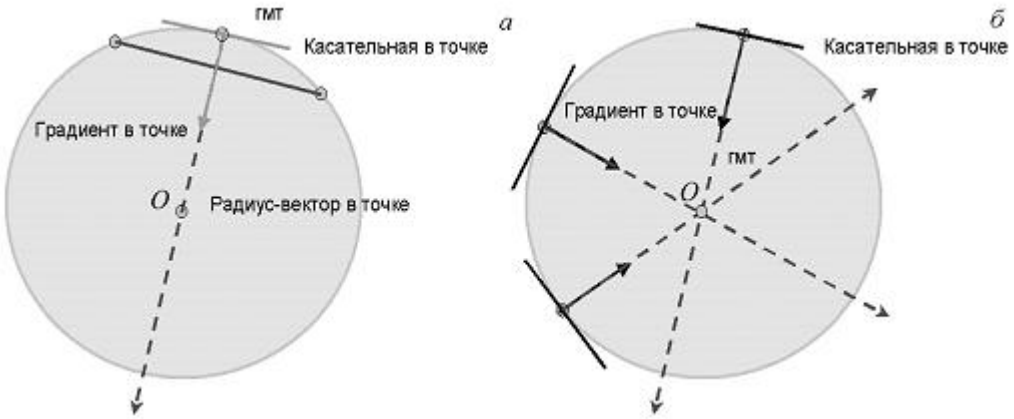


Преобразование Хафа

Аналогично – для окружностей

Обычно предварительно находят края

$$(x-x_0)^2+(y-y_0)^2=R^2$$



Преобразование Хафа

```
int main()
{
    cv::Mat image = cv::imread ("thined.jpg",1), greyscale;
    cvtColor( image, greyscale, CV_BGR2GRAY );

    cv::Matlb binary(image.size());
    cv::threshold( greyscale, binary, 127, 255, cv::THRESH_BINARY | cv::THRESH_OTSU );

    cv::vector<cv::Vec4i> lines;
    cv::HoughLinesP (binary,            //  Входной массив
                     lines,             //  Выходной массив
                     2,                 //  Шаг аккумулятора по радиусу
                     CV_PI/180/4,       //  шаг по углу
                     50,                //  пороговое значение
                     20,                //  МИНИМАЛЬНАЯ длинна линий
                     10);               //  МАКСИМАЛЬНЫЙ возможный разрыв между элементами
    std::cout << lines.size() << " lines found" << std::endl;
    for (int i=0; i<lines.size(); i++)
    {
        cv::line (image, cv::Point(lines[i][0], lines[i][1]), cv::Point(lines[i][2], lines[i][3]), cv::Scalar(0,0,255), 1, CV_AA);
    }
    cv::imshow("Lines", image);
    cv::waitKey();
    return 0;
}
```

Преобразование Хафа